CSCI 340 - Operating Systems I
Spring 2023

Collaboration Policy: Individual Assignment
Total points: 100 Points

1 Objective

Develop a C program that will allow the student to gain experience in the following areas:

- **Unix Development**: editing, compiling, executing, and debugging C programs, and the use of Makefiles
- **Basic Programming Concepts**: variable declaration, data types, arrays, pointers, operators, expressions, selection statements, looping statements, functions, and header files
- **File Input/Output (IO)**: reading data from a text file and displaying information to the console
- **Structures**: using the definition of a struct data type
- **Dynamic Memory**: allocating space on the heap and using pointers to structs

2 Provided Files

The four files listed below are provided to you:

- **lines.txt**: Each line in this text file defines a mathematical line. Specifically, a line is defined by two points (x_0, y_0) and (x_1, y_1), where $x_i, y_i \in R$. The format of the file is very simple, each line contains four decimal numbers, each separated by a comma. You will need to read each line from this file **up to** TOTAL_LINES number of lines (see file hw1.c for definition)
- **utils.h**: Header file that defines a line and a point struct used in this assignment, as well as the function prototypes to be completed by you. NOTE: You may not add new function declarations to this header file.
- **utils.c**: The file containing the implementations of the functions listed in utils.h. Having a different file for the implementation separates interface (the include file) from the implementation (the .c file).
- **hw1.c**: Source code file that includes a "stubbed out" version of the main function, and defines the libraries and constants used in this assignment. Please note: You may not remove, modify, or add additional library includes. The ones that are provided, are the only libraries allowed for this assignment.

Please read the comments carefully and follow the instructions. The *hw1.c*, *utils.h* and *utils.c* files contain many comments that provide:

1. basic definitions
2. specific calculations
3. restrictions on library function usage
4. basic step-by-step instructions

3 Todo

Using the *utils.h* file, you must provide working implementations within the corresponding *utils.c* file for the following function prototypes:

1. `int read_lines(char* filename, line_t* line_arr, int n)`
2. `double calc_direction(line_t* line_ptr)`
3. `double calc_magnitude(line_t* line_ptr)`

For each function prototype listed above, numerous comments are provided in the header file to guide you in this assignment. Please read them carefully, they either provide basic step-by-step instructions, or basic calculations.

In the *hw1.c* file, you must fully implement the main function, ie., `int main(int argc, char** argv)`. Please read the comments carefully. They provide the steps to be coded by you in the main method, and how the output results are to be displayed in the console.

4 Collaboration and Plagiarism

This is an individual assignment, ie. **no collaboration is permitted**. Plagiarism will not be tolerated. Submitted solutions that are very similar (determined by the instructor) will be given a grade of zero. Please do your own work, and everything will be OK.

5 Submission

Create a compressed tarball, ie. *tar.gz*, that only contains the completed *hw1.c*, *utils.h*, *utils.c* and *Makefile* files. The name of the compressed tarball must be your last name. For example, *leclerc.tar.gz* would be the correct name for me. You can easily create such a tarball by typing *make tarball*. Only assignments submitted in the correct format will be accepted (no exceptions). Submit the compressed tarball on Oaks in the Assignment/Dropbox folder for this assignment. You may resubmit the compressed tarball as many times as you like; I will only grade the newest submission.

Late assignments will not be accepted. Exceptions will only be made for extenuating circumstances, ie. death in the family, health related problems, etc. The due date is listed on Oaks. Do not email the assignment after the due date, it will not be accepted.

I'm happy to answer questions about the assignment, but I cannot tell you how to code the solution. Additionally, code debugging is your job. You may use the debugger (gdb) or print statements to help understand why your solution is not working correctly.

6 Grading Rubric

For this assignment the grading rubric is provided in the table shown below:

read_lines function implementation	40 points
calc_direction function implementation	10 points
calc_magnitude function implementation	10 points
main() function implementation	40 points

NOTE: If the program does not compile, I will try and fix the first syntax error or two. After that point, I will stop grading and inspect the code and assign a grade that likely is only a small fraction of the total points. Why? Because much (perhaps most) of the development process has not been performed by you. The act of testing and debugging are very important steps in the development process.

If your program compiles and runs, but it "segfaults", or crashes in some other way. Similarly, in this situation, I will try to "fix" one or two such occurences in your code. After that point, I will inspect the remaining code and assign a partial grade to it

In short, it is difficult to assess a grade for an assignment that does not ultimately run. In any case, please stop by after I have issued an assessment of your code if you feel there is more to it. I will require you to walk me through the code and convince me that indeed there are more points to be given.